Machine Learning

CSE 343/543

Lecture 2

Empirical Risk Minimization

Outline

- Learning Machines
- Risk Functional
- Calculus of variations
- Empirical Risk Minimization

Learning Machines

- Described through three components
  - Generator of random vectors *x*, i.i.d. from a fixed but <u>unknown</u> distribution F(*x*)
  - A supervisor (oracle/jyotish) which returns an output vector *y*, for every input vector *x*, as per the conditional distribution F(*y*|*x*), also fixed but <u>unknown</u>.
  - A learning machine capable of implementing a set of functions

$$f(\boldsymbol{x}, \boldsymbol{w}), \quad \boldsymbol{w} \in \Omega$$

- The learning problem is to choose from the given set of functions the one which best approximates the supervisor's response.
  - The selection is based on training samples $(x_i, y_i), \quad i = 1, \ldots, \ell$

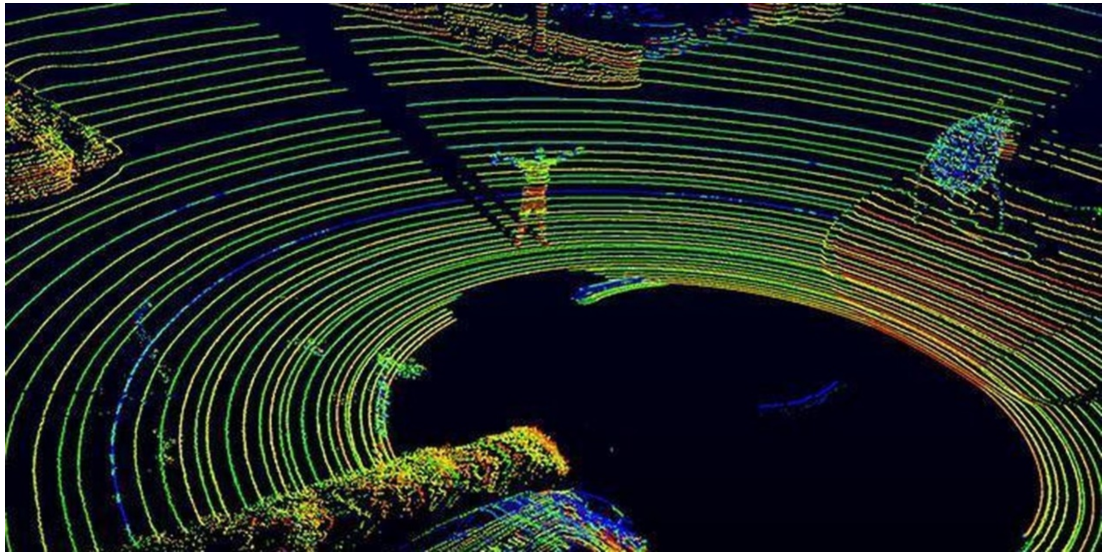
Vapnik, "Principles of Risk Minimization for Learning Theory", NIPS 1992

Regression: Concrete Example

LIDARs and Point Cloud

- Light Detection And Ranging (LiDaR)
  - Laser based range (distance) sensing

- Column of lasers sitting on a spinning motor



- Generates “Point Cloud” Data

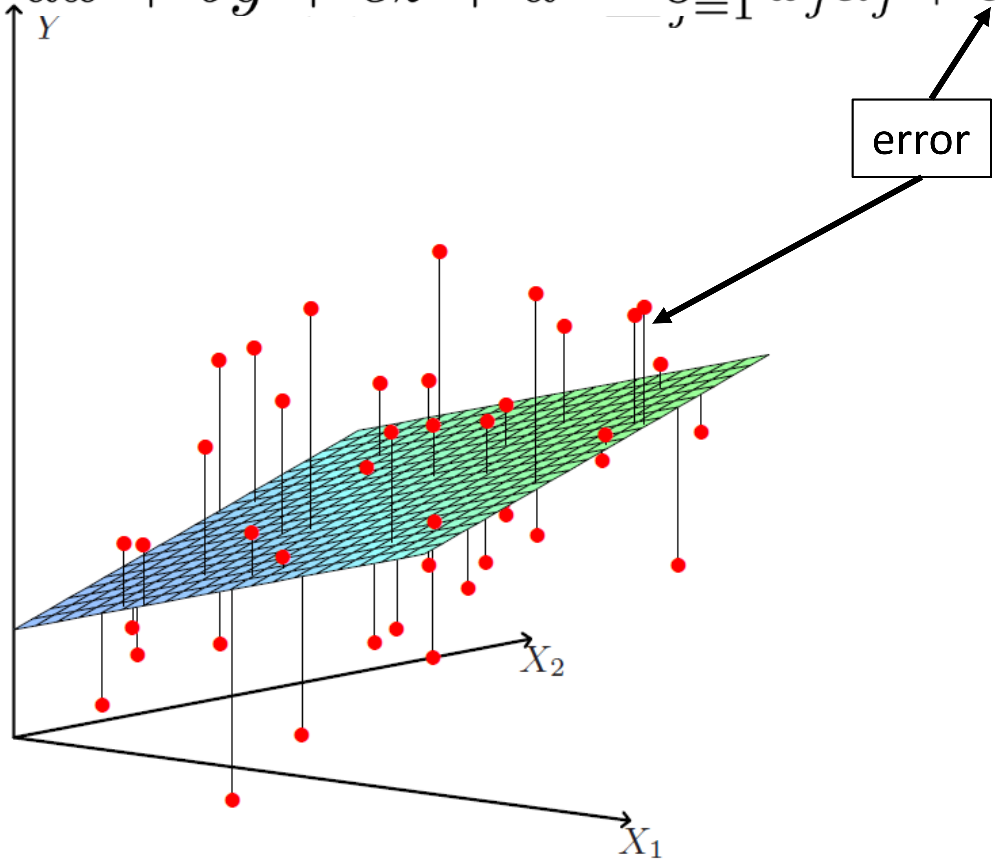
- Problem: Find Road Surface
  - Assume planarity

Regression: Concrete Example

- Find Road Surface
  - Assume planarity
- Given the points, estimate parameters Data/Feature
  - dimension (p=2) $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$
  - # training samples $(x_1, y_1) \ldots (x_N, y_N)$
  - parameters $\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_p)^\top$
- Evaluation Metric
  - How good is the fitted plane?

$$\text{Total Error} = \sum_{i=1}^{N} L(y, f(x, \alpha)) = \sum_{i=1}^{N} (y_i - f(x_i))^2$$
$$= \sum_{i=1}^{N} \left(y_i - \alpha_0 - \sum_{j=1}^{p} x_{ij}\alpha_j \right)^2$$

$ax + by + cz + d = 0$



Very Important!!

Regression: Abstraction

- What are Regressors and how are they modelled?
  - Essentially a function mapping
- Regression
  - Predict [avg. enrolment in 2018, CGPA] based on
    [current enrolment, grade, job offer, package]
  - Predict [mutual fund value] based on
    [stock prices, inflation, PM's foreign visit expenses]

$\alpha \in \Lambda$

$$f(x, \alpha) : \mathbb{R}^n \rightarrow \mathbb{R}^m$$

Some parameters governing the function *f*.
Can be abstract parameters like:
Degree of polynomial + coefficients

Loss functions

- To chose the best function, it makes sense to **minimize** a loss (or cost or discrepancy) between the response of the supervisor and the learning machine, given an input *x*

$$L(y, f(x, \alpha))$$

- Since we want to minimize the loss over *all* samples, we are interested in minimizing the *expected* loss

$$R(\alpha) = \int L(y, f(x, \alpha)) dF(x, y)$$

Loss Functions - Regression

p^{th} norm, p>=1

$$||x||_p = \left(\sum_{i=1}^{d} |x_i|^p\right)^{1/p}$$

- Least squares (L_2-norm) minimization
  - y takes continuous values

$$L(y, f(x,\alpha)) = ||y - f(x,\alpha)||_2^2$$

- L_1-norm minimization

$$L(y, f(x,\alpha)) = ||y - f(x,\alpha)||_1$$

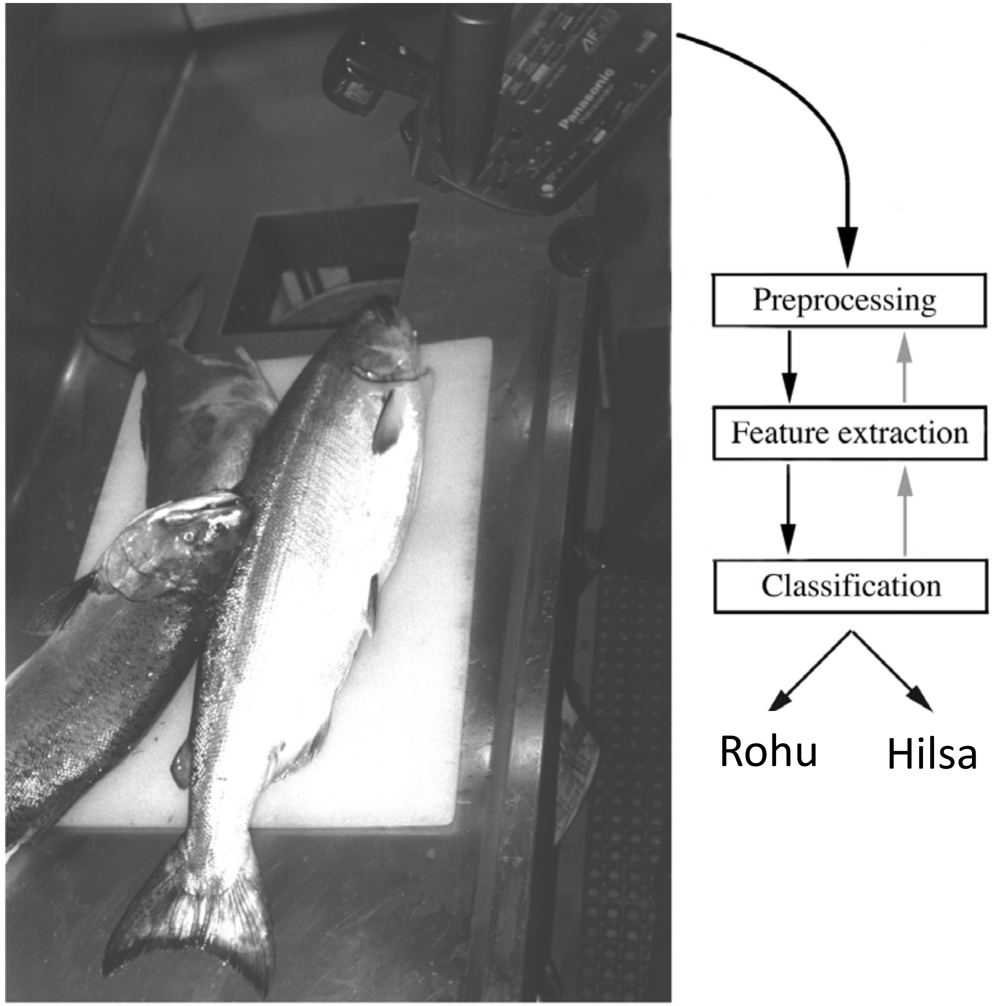
- Other example:
  - Huber loss (robust regression)

Classification: Concrete Example

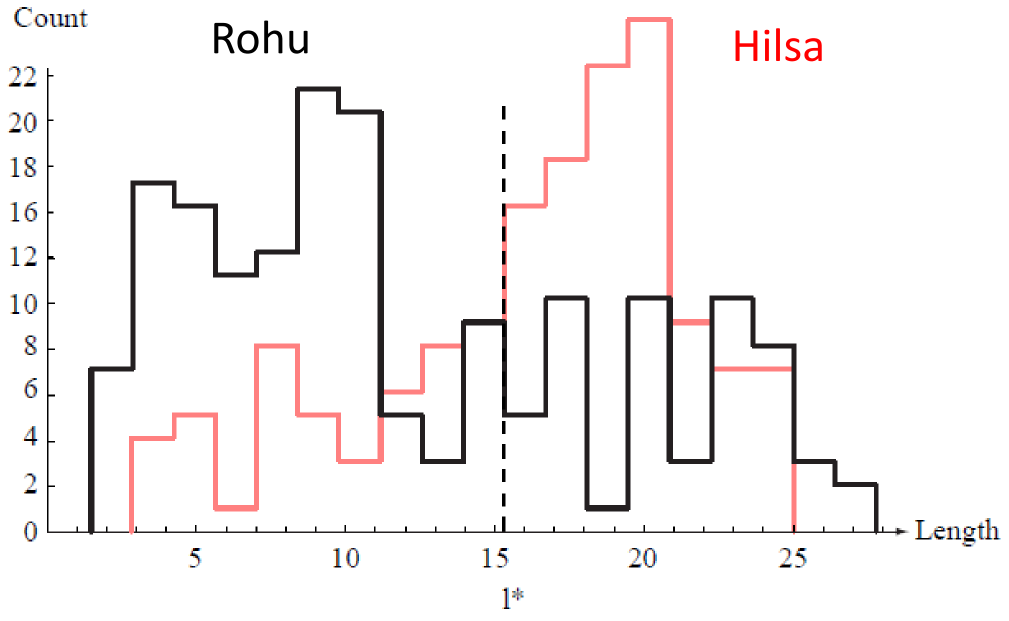
Classification: Concrete Example

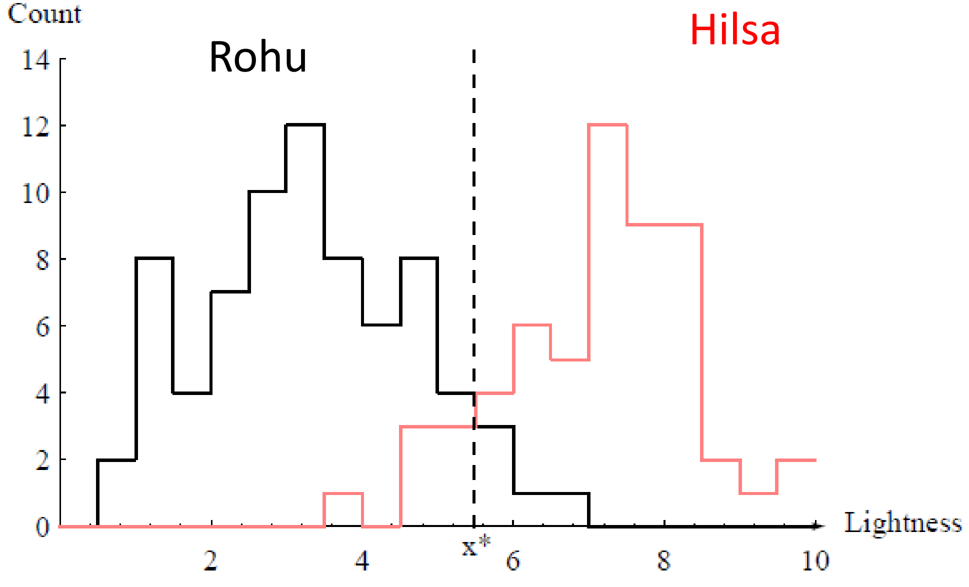
- Say you are ‘Laloo’, trying to make a good impression on ‘Didi’
  - You need to know your fish, but you don’t!
  - So you seek help from ML experts@IIITD to distinguish between
    ‘Rohu’ and ‘Hilsa’

- ‘Features’ for classification
  - Length/width
  - brightness

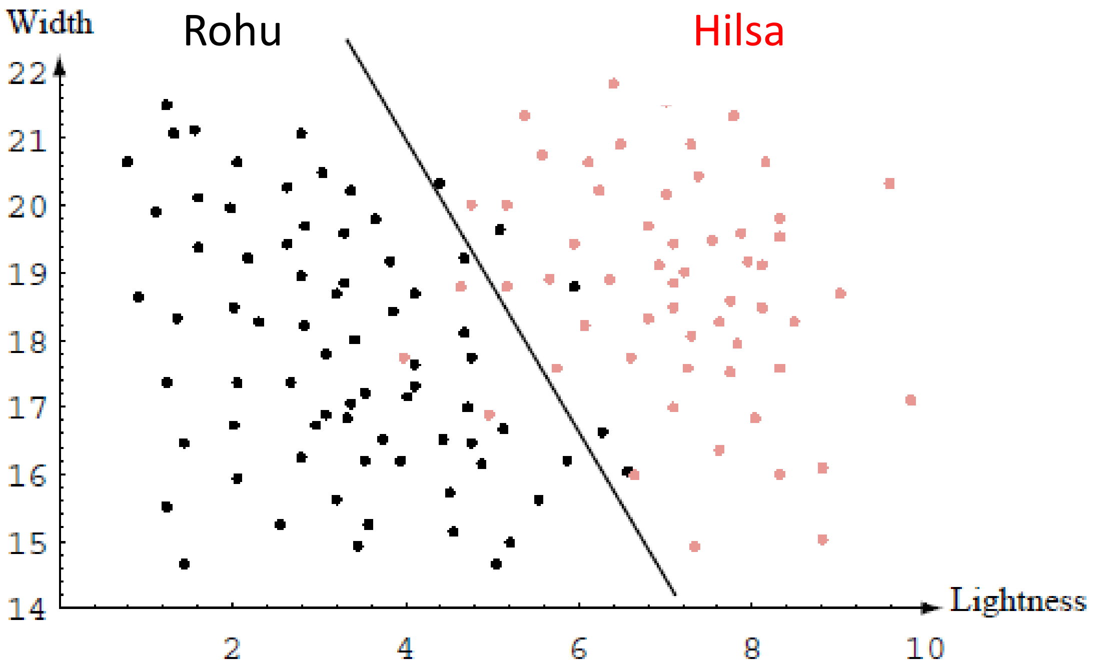


Classification: Rohu vs. Hilsa

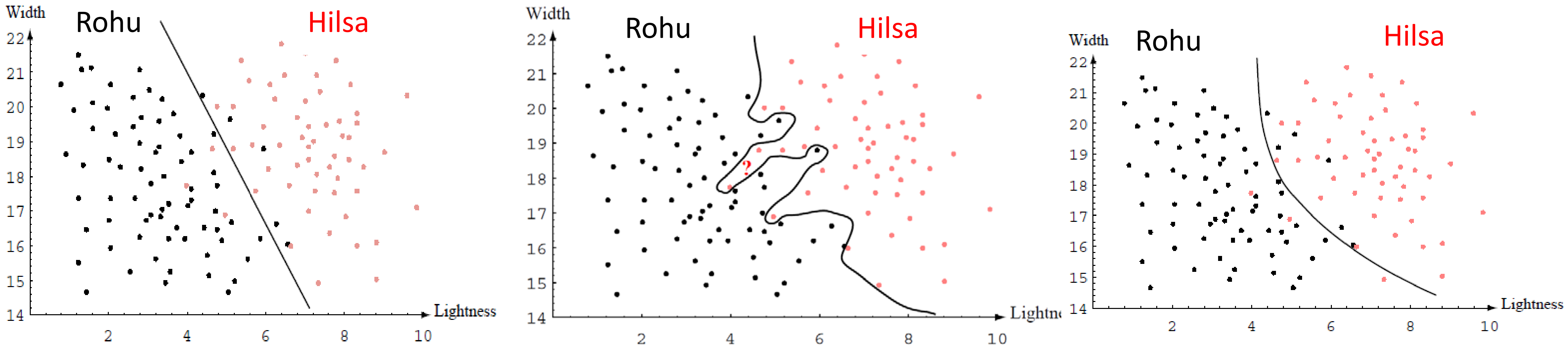




Classification: Rohu vs. Hilsa



Model Complexity: A note



Too Simple?

Too Complex?

Just Right?

Loss Functions - Classification

- Binary Classification with equal weights on misclassification
  - Minimize a 0-1 loss

$$L(y, f(x,\alpha)) = \begin{cases} 0 & \text{if } y = f(x,\alpha) \\ 1 & \text{if } y \neq f(x,\alpha) \end{cases}$$

- Classification with unequal weights on misclassification
  - Minimize a 0-10^7-500 loss

$$L(y, f(x,\alpha)) = \begin{cases} 0 & \text{if } y = f(x,\alpha) \\ 10^7 & \text{if } y = 1, f(x,\alpha) = 0 \\ 500 & \text{if } y = 0, f(x,\alpha) = 1 \end{cases}$$

Classification: Abstraction

- What are Classifiers and how are they modelled?
  - Essentially a function mapping
- Binary Classification
  - Detection (Spam/no Spam; bomb/no bomb)

$$f(x, \alpha) : \mathbb{R}^n \rightarrow \{-1, 1\}$$

- Multi-class Classification
  - ADAS[1] (pedestrians, vehicles, barricades,...)
  - Biometric system (Saket, Lokender, Sharat,...)

$$f(x, \alpha) : \mathbb{R}^n \rightarrow \{0, 1, 2, \ldots, k\}$$

$$\alpha \in \Lambda$$

Some parameters governing the function *f*.
Can be abstract parameters like:
one or several thresholds
one or several boundaries (linear/nonlinear)
No. of neurons + weights

[1]ADAS: Advanced Driver Assistance System

Risk Minimization

- The goal is to find the minimizer of the ***risk functional***

$$R(\alpha) = \int L(y, f(x,\alpha))dF(x,y), \quad \alpha \in \Lambda$$

i.e., find $f(x, \alpha^*)$ that minimizes

$$R(\alpha) \text{ over the class of functions } f(x,\alpha),\ \alpha \in \Lambda$$

where the joint PDF F(x,y) is unknown and the only available information is contained in the training set, i.e.,

$$(x_i, y_i), \quad i = 1, \ldots, \ell$$

Calculus of Variations

function

- y(x) as an operator that for any input x, returns a value y.

  e.g.: $y = mx + c$; $y=x^2$; $y=x^3$;

- Calculus used for function minimization over a space of <u>output values</u>

functional

- F[p] as an operator that for any input **<u>function</u>** p(x), returns a value F

  e.g. Entropy

  $$H[p] = -\int p(x) \ln p(x) dx$$

  Risk $R(\alpha)$

- Calculus of variations used for minimizing functionals over a space of <u>functions</u>

Bishop, Appendix D

Empirical Risk Minimization Principle

$$R(\alpha) = \int L(y, f(x, \alpha))dF(x, y), \quad \alpha \in \Lambda$$

- The risk functional is replaced by the *empirical risk functional*

$$R_{emp}(\alpha) = \frac{1}{\ell} \sum_{i=1}^{\ell} L(y_i, f(x_i, \alpha))$$

- Learning theory asks the following questions
  - What are the conditions for consistency?
  - How fast is the convergence rate?
  - How can one control generalization ability (on unseen examples from F(x,y))?
  - How can one construct algorithms to control generalization ability?

Loss Functions: A Probabilistic View

$$\text{Error} = L(y, f(x,\alpha)) = \sum_{i=1}^{N} (y_i - f(x_i,\alpha))^2$$

$$= \sum_{i=1}^{N} \left(y_i - \alpha_0 - \sum_{j=1}^{p} x_{ij}\alpha_j \right)^2$$

$$||\widehat{\epsilon}||_2^2 = ||y - \alpha^\top \widetilde{x}||_2^2$$

$$f(\boldsymbol{x}_i, \alpha) = \alpha^\top \boldsymbol{x}_i$$

$$f(\boldsymbol{x}_i, \alpha) = \alpha_1 x_{i1} + \alpha_2 x_{i2} + \ldots$$

Linear in parameters

$$\text{where, } \widehat{\epsilon} = [\epsilon_1, \epsilon_2, \ldots, \epsilon_N]^\top,\ y = [y_1, y_2, \ldots, y_N]^\top,\ \widetilde{x} = [x_1, x_2, \ldots, x_N]^\top$$

- If we model the noise as a zero mean Gaussian random variable with variance σ^2, the distribution is:

$$p(\epsilon_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\epsilon_i^2}{2\sigma^2} \right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{y_i - \alpha^\top x_i}{2\sigma} \right)^2$$

Loss Functions: A Probabilistic View

- Assuming i.i.d. errors, the joint probability of $p(\epsilon) = p(\epsilon_1, \epsilon_2, \ldots, \epsilon_N)$

$$p(\epsilon) = p(\epsilon_1, \epsilon_2, \ldots, \epsilon_N) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\epsilon_i^2}{2\sigma^2}\right)$$

$$= \frac{1}{(\sqrt{2\pi\sigma})^N} \exp \frac{-1}{2}\left(\frac{\sum_{i=1}^{N}(y_i - \alpha^\top x_i)^2}{\sigma^2}\right)$$

$$= \frac{1}{(\sqrt{2\pi\sigma})^N} \exp \frac{-1}{2}\left(\frac{||y - \alpha^\top \widetilde{x}||}{\sigma}\right)^2$$

- We can view this joint probability as a function of the parameters

$$\boxed{\ell(\alpha) = p(\epsilon_1, \epsilon_2, \ldots, \epsilon_N|\alpha) = \frac{1}{(\sqrt{2\pi\sigma})^N} \exp \frac{-1}{2}\left(\frac{||y - \alpha^\top \widetilde{x}||}{\sigma}\right)^2}$$

This is the likelihood function

Maximum Likelihood

- Maximize the likelihood over all available samples

$$\widehat{\alpha} = \arg\max_{\alpha} p(\epsilon_1, \epsilon_2, \ldots, \epsilon_N | \alpha) = \arg\max_{\alpha} \frac{1}{(\sqrt{2\pi}\sigma)^N} \exp\left(-\frac{||y - \alpha^\top \widetilde{x}||}{2\sigma}\right)^2$$

- Since log is a monotonic function, often log-likelihood is used

$$\widehat{\alpha} = \arg\max_{\alpha} \log p(\boxed{\epsilon_1, \epsilon_2, \ldots, \epsilon_N} | \alpha) = \arg\max_{\alpha} \left(-\sum_{i=1}^{N} (y_i - \alpha^\top x_i)^2 + \text{const}\right).$$

Measurements/data Parameters/model

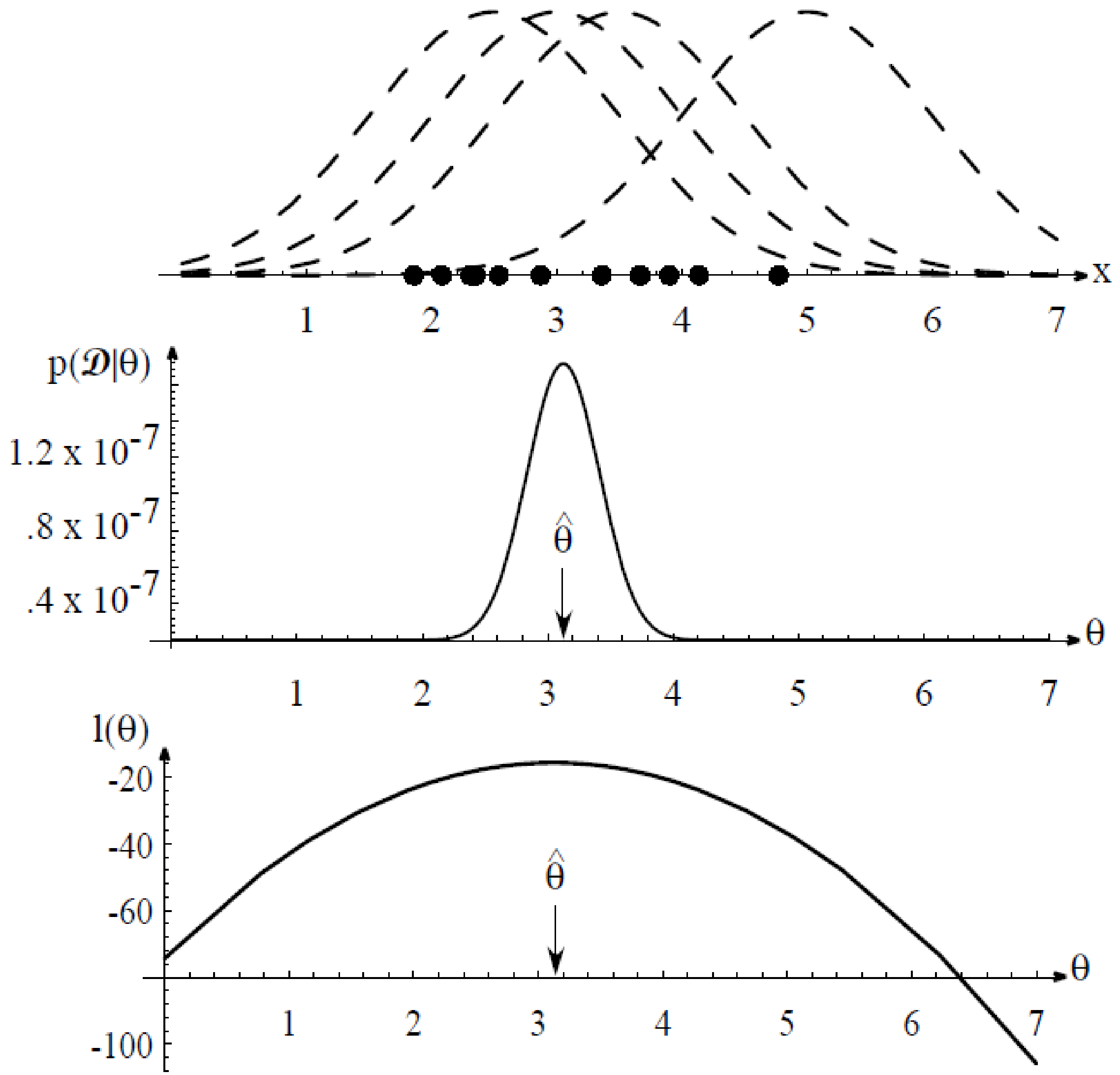
Maximum (log-)Likelihood

- Valid for an **arbitrary distribution**

$$\widehat{\alpha} = \arg\max_{\alpha} \log p(\epsilon_1, \epsilon_2, \ldots, \epsilon_N | \alpha)$$

$$= \arg\max_{\alpha} \sum_{i=1}^{N} \log p(\epsilon_i | \alpha)$$

Visualization: Maximum Likelihood



2